BLAST Search NCBI Nextgen Data Locally

A step-by-step demonstration of using sratoolkit for local BLAST search against NCBI SRA data

https://trace.ncbi.nlm.nih.gov/Traces/sra/sra.cgi?view=software

National Center for Biotechnology Information • National Library of Medicine • National Institutes of Health • Department of Health and Human Services

Introduction

The Sequence Read Archive (SRA) database at NCBI archives next generation (nextgen) sequence data submitted by the biomedical research community. NCBI also made most of the nextgen data from SRA accessible from the Web BLAST service. However, the rapidly increasing volumes of SRA data and the constraint of computational resources available make the performance of this SRA data access through BLAST uneven.

The standalone sratoolkit package from NCBI offers a viable and more reliable alternative. Using tools from this package, we can retrieve the desired dataset using prefetch, then BLAST search the downloaded dataset using the blastn_vdb (for nucleotide query) or tblastn_vdb (for protein query). We can retrieve matched hits from the target dataset using fastq-dump with the SRR accession and spot ID as input. This approach works best for Linux and Mac OSX platforms. For PC running Windows 10, the work around is to install a free Linux Subsystem APP, then install the sratoolkit under the Linux environment under the APP.

This handout addresses the installation and configuration of the sratoolkit under a CentOS Linux system (using a user's home directory that runs bash), demonstrates the downloading of example datasets from SRA, BLAST search against the downloaded datasets, and retrieval of sample hits from these datasets. We are using the home directory as the working directory.

Download and Installation of the sratooklit from NCBI

The latest version of the sratoolkit are available from the SRA software download page:
https://trace.ncbi.nlm.nih.gov/Traces/sra/sra.cgi?view=software

Right click the file name for the Cent OS non-sudo package, and copy the link for use with wget:

```
$ wget -o sratoolkit.tar.gz “https://ftp-trace.ncbi.nlm.nih.gov/sra/sdk/2.10.9/sratoolkit.2.10.9-centos_linux64.tar.gz”
```

The command saves the archive to a file named by the -o argument (-o sratoolkit.tar.gz).

Use tar utility to decompress and extract the downloaded archive to install it:

```
$ tar zxvpf sratoolkit.tar.gz
```

This inflates the compressed archive, then extracts the sratoolkit package in a new subdirectory. The following command shows the installed subdirectory:

```
$ ls -ltr | grep sra
drwxrwxr-x 5 tao sdesk     4096 Dec 15 12:00 sratoolkit.2.10.9-centos_linux64
```

Use ls command to see what are in the installed subdirectory:

```
$ ls -l sratoolkit.2.10.9-centos_linux64/
total 56
-rw-r--r-- 1 tao sdesk 24372 Dec 15 11:59 CHANGES
-rw-rw-r-- 1 tao sdesk  5641 Feb  2  2015 README-blastn
-rw-rw-r-- 1 tao sdesk  4975 Dec  4  2017 README-vdb-config
-rw-rw-r-- 1 tao sdesk  3792 Jul 21  2020 README.md
drwxrwxr-x 3 tao sdesk  4096 Dec 11 14:25 bin
drwxrwxr-x 3 tao sdesk  4096 Oct 11  2016 example
drwxrwxr-x 8 tao sdesk  4096 Dec 10 16:32 schema
```

Actual programs are in the /bin subdirectory with the release version suffix to make the program name. Program names without release version suffix are also available to help avoid breaking users' custom workflow, they are soft links to the actual program.

For this demonstration, we only uses the following programs:

```
vdb-config
prefetch
blastn_vdb
tblastn_vdb
fastq-dump
```

Configuration of Installed sratooklit

Modify PATH environment variable

Before configuring the installation, first modify the PATH variable by append the path to the sratoolkit’s bin subdirectory so the system knows where to look for the invoked program:

```
$ export PATH=$PATH:~/sratoolkit.2.10.9-centos_linux64/bin
```

A bash command to export a variable called PATH, by set it to existing value ($PATH), with append (:) and additional path (~/sratoolkit.2.10.9-centos_linux64/bin, where ~ marks the home directory).

Configure the sratoolkit

This step specifies where the installed sratoolkit’s programs should look for to store downloaded SRA datasets, and where blast programs should look for datasets locally. The configuration goes through the vdb-config program through in interactive text menu -drive interface, to be explained through text description and screenshots.

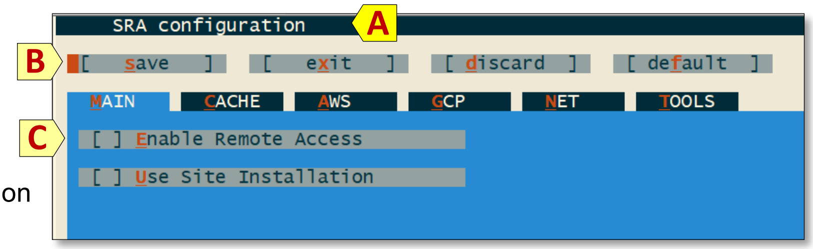
```
$ vdb-config -i
```

This launches a DOS-like window (**A**). There the active item is marked by red rectangle (**B**). Use tab key to cycle through different menu options. Alternatively, use red letter or mouse click to jump to specific options.

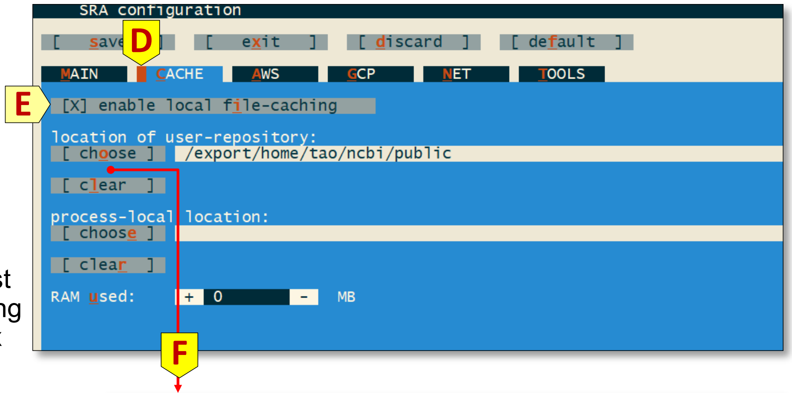
Specific setting changes:

1) Disable remote access. Since it is unreliable for blast to fetch the data during the search, disable this by typing “e” to toggle off the “Enable Remote Access” checkbox (i.e., remove the X, **C**).

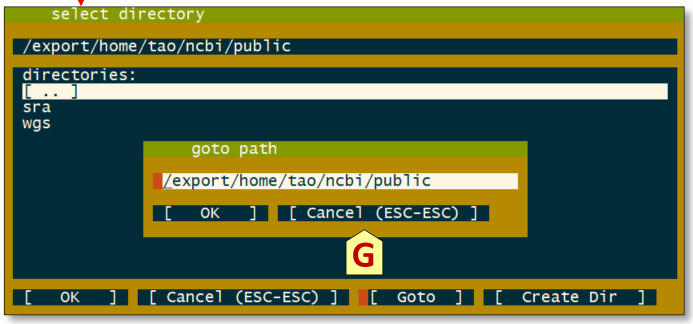
2) Enable local file-caching. Type “c” to go to the Cache menu (**D**), and toggle on “enable local file-caching” by typing “l” (i.e., enable the X, **E**).



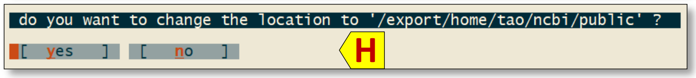
```
SRA configuration
[  save  ]   [  exit  ]   [ discard  ]   [ default  ]
MAIN   CACHE   AWS   GCP   NET   TOOLS
[ ] Enable Remote Access
[ ] Use Site Installation
```



```
SRA configuration
[  save  ]   [  exit  ]   [ discard  ]   [ default  ]
MAIN   CACHE   AWS   GCP   NET   TOOLS
[X] enable local file-caching

location of user-repository:
[ choose ]   /export/home/tao/ncbi/public

[ clear  ]

process-local location:
[ choose ]

[ clear  ]

RAM used:   + 0      -   MB
```



```
select directory
/export/home/tao/ncbi/public
directories:
[ .. ]
sra
wgs

        goto path
        /export/home/tao/ncbi/public
        [   OK   ]  [ Cancel (ESC-ESC) ]

[   OK   ]  [ Cancel (ESC-ESC) ]  [  Goto  ]  [  Create Dir  ]
```



```
do you want to change the location to '/export/home/tao/ncbi/public' ?
[  yes  ]  [  no  ]
```

3) Edit the Cache directory if needed. Type “o” to go to the “choose” line, hit enter to invoke the select directory prompt (**F**). Tab to “Goto” option and hit enter to edit the path in the the “goto path” prompt (**G**). Tab to “OK” and hit enter to return to previous prompt, confirm the change when prompted (**H**).

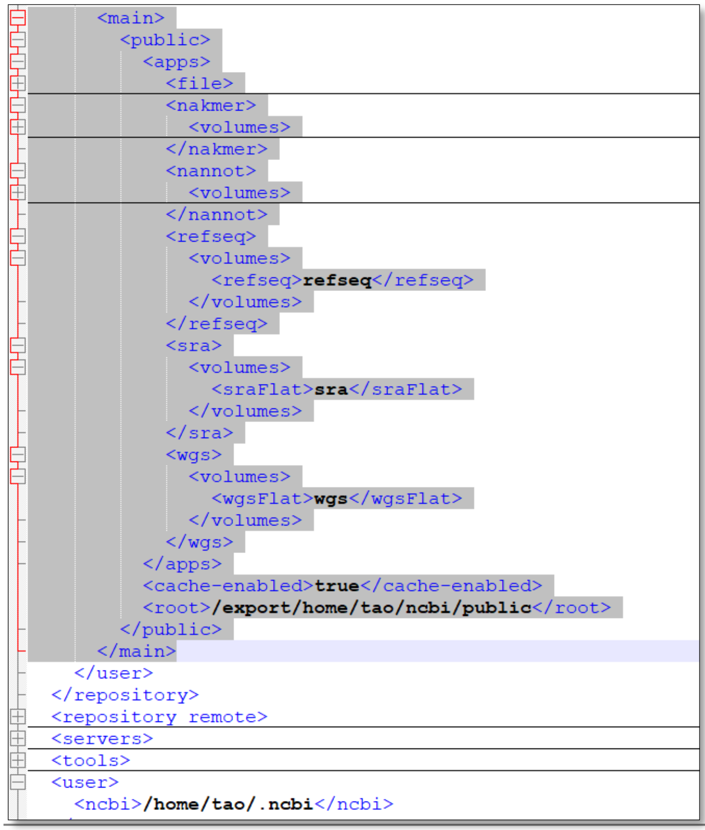
4) Save the changes. Type “s” to save, then “x” to exist the configuration.

Examine the existing configuration

Call vdb-config to generate the configuration xml:

```
vdb-config > my_vdb_settings.xml
```

Use an xml file editor to interactively examine the settings as shown (left, local file storage section).



```
      <main>
        <public>
          <apps>
            <file>
            <nakmer>
              <volumes>
            </nakmer>
            <nannot>
              <volumes>
            </nannot>
            <refseq>
              <volumes>
                <refseq>refseq</refseq>
              </volumes>
            </refseq>
            <sra>
              <volumes>
                <sraFlat>sra</sraFlat>
              </volumes>
            </sra>
            <wgs>
              <volumes>
                <wgsFlat>wgs</wgsFlat>
              </volumes>
            </wgs>
          </apps>
          <cache-enabled>true</cache-enabled>
          <root>/export/home/tao/ncbi/public</root>
        </public>
      </main>
    </user>
  </repository>
  <repository remote>
  <servers>
  <tools>
  <user>
    <ncbi>/home/tao/.ncbi</ncbi>
```

Searching against SRA Datasets

The demonstration consists of two use cases. The queries will be the cDNA sequence (XM_001421454) and its encoded protein (XP_001421491) of a theoretical fatty acid synthase. The goal is to find related genomic reads matching to the input cDNA, and the RNA-seq reads for the protein from closely related species.

The genomic and RNA-seq reads are from SRA searches https://go.usa.gov/xsd83 and https://go.usa.gov/xsd8a, respectively. Steps to batch process the list using EntrezDirect to get the SRR accessions is in the note.

Use Case 1: Finding the genomic reads for the input cDNA query

1. Download the SRR runs to local storage

Copy and paste the following SRR accessions into a text file named srr_list.txt:

```
SRR4026730
SRR4026187
```

Invoke the prefetch to download:

```
$ prefetch --option-file srr_list.txt
```

which will produces console output similar to :

```
2021-03-11T18:38:45 prefetch.2.10.9: 1) Downloading 'SRR4026730'...
2021-03-11T18:38:45 prefetch.2.10.9:  Downloading via HTTPS...
2021-03-11T18:39:23 prefetch.2.10.9:  HTTPS download succeed
2021-03-11T18:39:26 prefetch.2.10.9:  'SRR4026730' is valid
2021-03-11T18:39:26 prefetch.2.10.9: 1) 'SRR4026730' was downloaded successfully
2021-03-11T18:39:26 prefetch.2.10.9: 'SRR4026730' has 0 unresolved dependencies
...
```

Use vdb-validate to validate the runs if desired.

```
$ vdb-validate --option-file srr_list.txt
```

Output is not shown.

2. Search the cDNA sequence against the downloaded SRR runs

Create a simple query file:

```
$ echo XM_001421454 > lipid_synthase_nt.acc
```

BLAST search the query against the download runs:

```
$ blastn_vdb -query lipid_synthase_nt.acc -db "SRR4026730 SRR4026187" -task dc-megablast \
   -outfmt 7 –max_target_seqs 2000 -out cdna_vs_genomic.txt
```

This calls blastn_vdb, use file as query (-query lipid_synthase_nt.acc), search against the two runs (-db "SRR4026730 SRR4026187"), with discontiguous megablast (-task dc-megablast), requests tabular output (-outfmt 7) for 2000 hits (-max_target_seqs 2000) to be saved in a file (-out cdna_vs_genomic.txt). For pairwise alignment, use -outfmt 0.

3. Retrieve hits with the reads' sequence id

The matched reads are marked by their sequence ids given in second column, such as this:

```
XM_001421454.1 SRA:SRR4026730.20433005.2    92.105  76     6      0      767    842    76     1.23e-21
       111
```

Invoke sam-dump to retrieve the reads. Paired layout will retrieve two reads that needs further cleanup.

```
$ fastq-dump -Z -I -N 20433005 -X 20433005 --split-files --fasta 70 ./ncbi/public/sra/SRR4026730.sra
```

This calls fastq-dump, and request console output (-Z), with reads id enabled (-I). It asks one spot at a time by using the same spot id input to min (-N) and max (-X) options, with reads split (--split-files), in fasta format (--fasta 70, 70 bases per line). The local file path with extension is required for reading from locally stored runs.

This output the following to the console, with only the .2 read being the one matching to input query.

```
Read 1 spots for ./ncbi/public/sra/SRR4026730.sra
Written 1 spots for ./ncbi/public/sra/SRR4026730.sra
>SRR4026730.20433005.1 20433005 length=76
ACAGAGCCGGCGGAGGTGTTGCCATACTCGGCAATGTTGCTAACTACCTTGTCTTCAGTCAAGCCAAATC
GCTGCG
>SRR4026730.20433005.2 20433005 length=76
CGTTTTGCAACATCTCCATGAACGGGCAAGACGTCTTCAAGTTTGCCGTGCGAACGGTCCCGATGACGGT
GAACAA
```

Searching against SRA Datasets (cont.)

Use Case 2: Finding cDNA reads for an input protein query

1. Download the SRR runs to local storage

Copy and paste the following SRR accessions into a text file named srr_rna_list.txt:

```
SRR7121159
SRR7121158
```

Invoke the prefetch to download:

```
$ prefetch --option-file srr_rna_list.txt
```

The console output is omitted.

Invoke vdb-validate to validate the download:

```
$ vdb-validate --option-file srr_rna_list.txt
2021-03-11T21:54:23 vdb-validate.2.10.9 info: Validating '/export/home/tao/ncbi/public/sra/
SRR7121159.sra'...
2021-03-11T21:54:23 vdb-validate.2.10.9 info: Database 'SRR7121159.sra' metadata: md5 ok
2021-03-11T21:54:23 vdb-validate.2.10.9 info: Table 'SEQUENCE' metadata: md5 ok
2021-03-11T21:54:23 vdb-validate.2.10.9 info: Column 'ALTREAD': checksums ok
2021-03-11T21:54:23 vdb-validate.2.10.9 info: Column 'QUALITY': checksums ok
2021-03-11T21:54:23 vdb-validate.2.10.9 info: Column 'READ': checksums ok
2021-03-11T21:54:23 vdb-validate.2.10.9 info: Database '/export/home/tao/ncbi/public/sra/SRR7121159.sra'
contains only unaligned reads
2021-03-11T21:54:23 vdb-validate.2.10.9 info: Database 'SRR7121159.sra' is consistent
...
```

2. Search the protein sequence against the downloaded SRR runs

Create a simple query file:

```
$ echo XP_001421454 > lipid_synthase_aa.acc
```

Search the query with tblastn_vdb against downloaded runs:

```
$ tblastn_vdb -query lipid_synthase_aa.acc -db "SRR7121159 SRR7121158" -num_descriptions 2000
  -num_alignments 2000 -outfmt 0 -out prot_vs_cdna.txt
```

Examine the result using shell command, sample Descriptions and Alignments sections are:

```
...
                                                                      Score     E
Sequences producing significant alignments:                          (Bits)  Value

SRA:SRR7121159.478097.2 478097                                        87.8    3e-20
…
>SRA:SRR7121158.4239950.1 4239950
Length=125

 Score = 87.8 bits (216),  Expect = 3e-20, Method: Composition-based stats.
 Identities = 39/41 (95%), Positives = 41/41 (100%), Gaps = 0/41 (0%)
 Frame = -1

Query  161  IGLVNGAQFIRAGGYKNVLVIGGDVLSRYVDWRDRGTCILF  201
            IGLVNGAQ+IRAGG+KNVLVIGGDVLSRYVDWRDRGTCILF
Sbjct  125  IGLVNGAQYIRAGGFKNVLVIGGDVLSRYVDWRDRGTCILF  3
```

3. Retrieve the hits from the locally stored runs

Use the same approach described in Use Case 1 to retrieve hits of interest. For example, those hits can be used to assemble the full-length CDS of the homolog from this closely related organism.

Refer to the note for additional information for additional information on batch retrieval of reads from a given a set of matches in tabular output, based on the output from these two use cases.

Advanced Setup

For large datasets, here are additional suggested ways to batch and streamline the steps described in these two used cases.

1. Use EntrezDirect (EDirect) to locate SRA runs of interest and extract the run accessions.
The best way to examine the available SRA datasets is again through web SRA searches (www.ncbi.nlm.nih.gov/sra), using Query Builder under the Advanced page. However, processing a large set manually is impractical.

EntrezDirect package for unix provides a convenient way for us automate this process. An example set of commands below demonstrates that. Replacing the query terms within quotes will process query-specified custom set.

```
$ esearch -db sra -query 'SRR448586 OR SRR448575 OR SRR448580 OR SRR448581' | \
  esummary | xtract -pattern DocumentSummary -PRJ Bioproject -SMPL Biosample \
  -element "&PRJ" "&SMPL" Run@acc Run@total_spots Run@total_bases > run_summary_info.txt
```

The above set of EDirect commands searches SRA with a set query terms in quote, passes the information to retrieve the esummary XML, then use xtract parser to pick up required information from each record. The output is redirected to a file for record keeping. It contains Bioproject, Biosample, Run, total spots, and total bases as shown below:

```
PRJNA144099    SAMN02953814   SRR448586      114636151      22927230200
PRJNA144099    SAMN02953814   SRR448581      100024835      20004967000
PRJNA144099    SAMN02953814   SRR448580      74910341       7341213418
PRJNA144099    SAMN02953814   SRR448575      124268011      24853602200
```

Extracting the third column to a new file creates an input file for prefetch for batch download:

```
$ cut -f3 run_summary_info.txt > run_acc_list.txt
```

More information on EDirect is online at www.ncbi.nlm.nih.gov/books/NBK179288.

2. Use a list of run accessions for batch download.
Prefetch takes an input file through the --option-file <file> argument. When the input file is a list of run accessions, it will try to download them all if there are enough space in the working directory. The following command does just that:

```
$ prefetch --option-file run_acc_list.txt
```

Which reports progresses to the console, as discussed in Use Case 1 earlier.

3. BLAST search against downloaded runs
The blastn_vdb program is for searching a nucleotide query directly against selected runs. The tblastn_vdb program is for search a protein query against selected runs translated in all six frames on the fly.

For small set of runs, the -db argument accepts a space-separated list of databases in quotes:

```
$ blastn_vdb -db "SRR448580 SRR448586 SRR448581 SRR409113 SRR409114" ...
```

For convenience, especially for a long list of runs or for frequently used set, a database alias file should be used instead. Such an alias file is basically a plain text file with defined format. It instructes BLAST programs from sratoolkit on what runs to search. An alias file must have the .nvl extension, with the following base format:

```
# Alias file template, created on Wed Mar 10 18:45:58 2021
TITLE Sample SRA database with multiple runs
VDBLIST SRR448580 SRR448586 SRR448581 SRR409113 SRR409114
NSEQ 827678676
LENGTH 75127012818
```

The lines start with # or field name in CAPITAL letters:

- **#** marks comment lines to provide reference information.
- **TITLE** is the title we give to the database.
- **VDBLIST** line contains a space-limited run accessions already downloaded locally.
- **NSEQ** line contains the number of sequences, reads more precisely for run datasets
- **LENGTH** is the total number of bases.

The input to the last two lines can be calculated from the EDirect output above. If you don't have them, leave the lines OUT altogether. If the file is saved as **test_srr_db.nvl**, call the database using the filename without the .nvl extension:

```
$ blastn_vdb -db test_srr_db ...
```

Advanced Setup (cont.)

4. Extract hits from the runs.

The best results format for parsing is the tabular output. For the following output generated by -outfmt 7 (Q.out), the second column contains the information for read extract from the locally stored runs. Note that columns in this output can be customized to include other information, type “blastn_vdb -help” to see the details.

```
$ cat Q.out
# BLASTN 2.10.1+
# Query: XM_001421454.1 Ostreococcus lucimarinus CCE9901 predicted protein partial mRNA
# Database: SRR4026730
# Fields: query acc.ver, subject acc.ver, % identity, alignment length, mismatches, gap opens, q. start, q.
end, s. start, s. end, evalue, bit score
# 10 hits found
XM_001421454.1 SRA:SRR4026730.20433005.2    92.105  76      6       0       767     842     1       76      6.31e-
22      111
...
# BLAST processed 1 queries
```

The following shell commands process it to split out column for use with fastq-dump to dump out the reads of interest.

```
$ grep -v "#" Q.out | cut -f 2 | \
  sed -r "s/SRA:(([DES]RR[0-9]+)\.([0-9]+)\.([0-9]+))/\1\t\3\t\3\t\2/;" > Q_match_read.txt
```

It greps out the hit lines, cut out the second column, then split out the relevant subfields and send the output to a file named Q_matth_read.txt in a tab-delimited format:

```
SRR4026730.20433005.2 20433005       20433005       SRR4026730
SRR4026730.19638698.2 19638698       19638698       SRR4026730
SRR4026730.18592619.2 18592619       18592619       SRR4026730
SRR4026730.17745360.1 17745360       17745360       SRR4026730
SRR4026730.17018021.2 17018021       17018021       SRR4026730
SRR4026730.13308509.2 13308509       13308509       SRR4026730
SRR4026730.11682094.2 11682094       11682094       SRR4026730
SRR4026730.11195287.2 11195287       11195287       SRR4026730
SRR4026730.4265430.2  4265430 4265430 SRR4026730
SRR4026730.580275.2   580275  580275  SRR4026730
```

The first column is for reference need to keep track matched reads, the second and third column are the same read id, and the last column, the run accession. It can be used as input to fastq-dump for read extraction:

```
$ cut -f 2,3,4 Q_match_read.txt  | xargs -n 3 sh -c 'fastq-dump -Z -I --split-files --fasta 70 -N $0 -X $1
ncbi/public/sra/$2.sra >>reads_extracted.fa'
```

It takes column 2-4 from the input and pass it to xargs for processing with fastq-dump by feeding column 2 to -N, column 3 to -X, and column 4 as run argument with path prefixed.

The reads will be directed and appended to the specified file, the console output is only for our information:

```
Read 1 spots for ncbi/public/sra/SRR4026730.sra
Written 1 spots for ncbi/public/sra/SRR4026730.sra
Read 1 spots for ncbi/public/sra/SRR4026730.sra
Written 1 spots for ncbi/public/sra/SRR4026730.sra
...
```

The fasta dump looks like this:

```
$ head -6 reads_extracted.fa
>SRR4026730.20433005.1 20433005 length=76
ACAGAGCCGGCGGAGGTGTTGCCATACTCGGCAATGTTGCTAACTACCTTGTCTTCAGTCAAGCCAAATC
GCTGCG
>SRR4026730.20433005.2 20433005 length=76
CGTTTTGCAACATCTCCATGAACGGGCAAGACGTCTTCAAGTTTGCCGTGCGAACGGTCCCGATGACGGT
GAACAA
```